AFRL-IF-RS-TR-2007-200
Final Technical Report
September 2007

RAPID COMMUNITY OF INTEREST (COI) INFOSPACES CREATION AND DEPLOYMENT USING KAOS AND CMAPS

Florida Institute for Human and Machine Cognition (IHMC)

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-IF-RS-TR-2007-200 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
JAMES HANNA Work Unit Manager	JAMES W. CUSACK Chief, Information Systems Division Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
SEP 2007	Final	Mar 06 – May 07

4. TITLE AND SUBTITLE

RAPID COMMUNITY OF INTEREST (COI) AND DEPLOYMENT USING KAOS AND CMAPS

5a. CONTRACT NUMBER
FA8750-06-2-0065

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62702F

6. AUTHOR(S)

Andrzej Uszok and Jeff Bradshaw

5d. PROJECT NUMBER
ICED

5e. TASK NUMBER
06

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Florida Institute for Human and Machine Cognition (IHMC)
40 S Alcaniz Street
Pensacola FL 32502

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/IFSE
525 Brooks Rd
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2007-200

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# AFRL-07-0088

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The important results of the project include a deepened understanding of the community of interest (COI) lifecycle, definition of requirements for tools supporting COI lifecycle and description of the COI lifecycle dataflow. Additionally, it was shown how consistent usage of ontology in the COI supporting tool can add significant flexibility and richness to the process. The developed COI-Tool has these main features:
- Capture and share COI configurations in two synchronized representations
- Unique user-friendly Cmap environment with integrated Web Search, simultaneous collaboration and version control
- Facilitation of the COI implementation through integration with OIM RI and KAos Policy Service
- Reuse of COI models

15. SUBJECT TERMS
Community of interest, ontology, policy, concept maps, OWL, IMS, KAos, Cmap Tools COE

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	UL	26	James Hanna
U	U	U			19b. TELEPHONE NUMBER (*Include area code*) N/A

Table of Contents

List of Figures

Executive Summary

Communities of Interest (COIs) are the means by which the strategy of net-centric information sharing is implemented. Unfortunately, the implementation of COIs has been hampered by a lack of methodologies and software tools to effectively support COI lifecycle spirals. The purpose of this project is threefold: 1) to broaden the understanding of the full lifecycle of the COI, 2) to develop a software prototype demonstrating how support for each element of the lifecycle might be provided, and 3) to integrate this prototype with the AFRL OIM RI in order to validate the approach and enhance IMS capabilities.

Specific objectives for the COI-Tool prototype include the following:

- Ability to model specific roles and physical resources for COI data producers and consumers so that each participant knows what is expected from them;
- Ability to define the semantics and structure of the COI information in a way that allows automatic encoding in a formal computer processable notation;
- Minimization of training and support requirements for COI and IMS personnel through simple user interfaces and automation of tedious aspects of lifecycle support;
- Mapping and translation of simple incompatibilities between semantic information from different sources.

The initial part of the project was devoted to understanding existing COI research, developing a detailed specification of requirements for each of the COI lifecycle phases, and determining how best to address each of these requirements. We assembled a sizeable collection of COI resources on the project Web page. From these resources and from discussions with Dr. Larry Warrenfeltz an IHMC expert and former commander at the Naval Oceanographic Office who has extensive experience in the creation of METOC COIs, we derived the requirements that guided the remainder of the project.

In order to address the semantic aspects of COI modeling we based our approach on the most widely-adopted standard today for semantically-rich model representation: the W3C's OWL (Web Ontology Language, http://www.w3.org/TR/owl-features). An exciting part of the results of this project was the first time integration and enhancement of two existing IHMC tools with the AFRL IMS (JBI RI 1.2.6) in order to create a comprehensive environment supporting the COI lifecycle. Numerous new mechanisms and capabilities had to been added to these tools to make this possible.

A prototype supporting all three phases of the COI lifecycle was developed. In order to test usability of the developed methodology an example METOC community of interest was selected and developed using the tool. The important results of the project include a deepened understanding of the COI lifecycle, definition of requirements for tools supporting COI lifecycle and description of the COI lifecycle dataflow. Additionally it was shown how consistent usage of ontology in the COI supporting tool can add significant flexibility and richness the process. Additional information about COI-Tool and other technical results of the project are available at the project Web site *http://ontology.ihmc.us/COI/*.

1. Project Overview

A Community of Interest (COI) has been defined as "a collaborative group of users who must exchange information in pursuit of their shared goals, interests, missions, or business processes and who therefore must have shared vocabulary for the information they exchange" (Guidance for Implementing Net-Centric Data Sharing, DoD 8320.02-G, 12 April 2006, p. 11). COIs are the means by which net-centric information sharing is implemented, and members "are responsible for making information visible, accessible, understandable, and promoting trust – all of which contribute to the data interoperability necessary for effective information sharing" (ibid).

For the purposes of this project we assume that the members of a given COI will have a common mission they are supporting and that there will be one person, the COI manager, who is responsible to make sure that the resources of the COI are adequate to fulfill that mission. To formally define such a COI, information about producers, consumers, data product schemas, and policies governing information sharing and any other interaction among parties must be modeled. In addition, roles and relationships that relate people, activities, and types of information must also be captured in a form that is easy for COI members to understand and use. This additional information includes things such as:

- COI scope
- types of information to be exchanged
- types of consumers
- infospace managers
- applications used by information consumers
- degree of information integration
- information security activities
- consensus set of vocabulary terms and definitions

The process of formally defining all these elements will help the COI manager determine whether the aggregated assets are adequate for performing the mission.

The implementation of COIs has been hampered by a lack of methodologies and software tools to effectively support COI lifecycle spirals (Fig. 1). The budding area of COI research has understandably focused on *understanding* the lifecycle and, in particular, the process of collaboration behind the creation of a common vocabulary. In support of such activities, the DoD has developed a Web-based Metadata Registry to enable schema sharing. The purpose of this project is threefold: 1) to broaden the understanding of the full lifecycle of the COI, 2) to develop a software prototype demonstrating how support for each element of the lifecycle might be provided, and 3) to integrate this prototype with the AFRL OIM RI in order to validate the approach and enhance IMS capabilities.

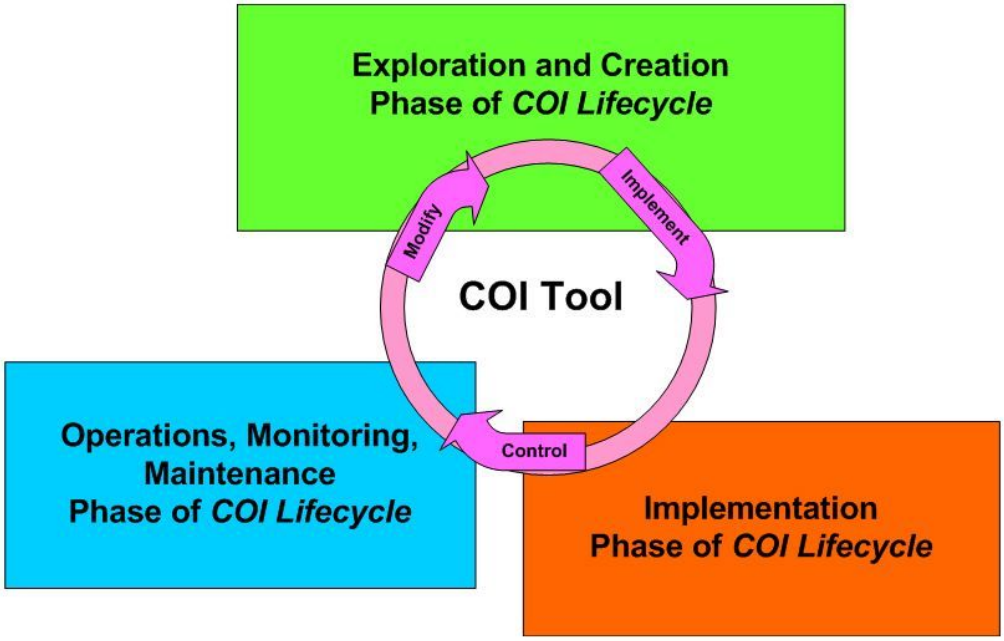


Figure 1: Major Phases of the COI Lifecycle Supported by the Prototype

Specific objectives for the COI-Tool prototype include the following:

- Ability to model specific roles and physical resources for COI data producers and consumers so that each participant knows what is expected from them;
- Ability to define the semantics and structure of the COI information in a way that allows automatic encoding in a formal computer processable notation;
- Minimization of training and support requirements for COI and IMS personnel through simple user interfaces and automation of tedious aspects of lifecycle support;
- Mapping and translation of simple incompatibilities between semantic information from different sources.

Requirements documents for the AFRL OIM RI (formerly JBI) listed the following objectives:

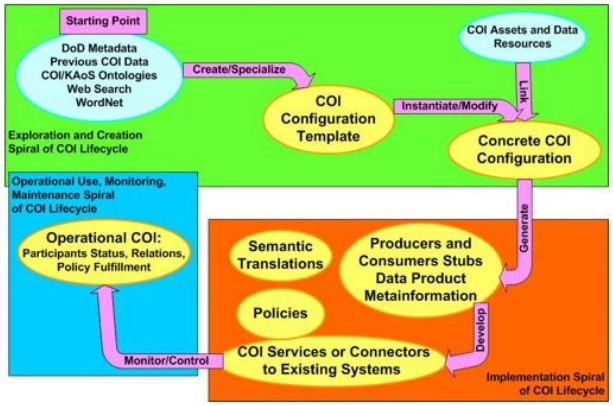
- Information exchange;
- Transformation of data;
- Distributed collaboration;
- Automatic incorporation of diverse units.

This project will focus on the last two objectives, which to date have not been as thoroughly addressed in previous research as have the first two.

A Quad Chart providing an overview of the project is given on the following page.

Rapid COI Infospaces Creation and Deployment using KAoS and COE

ihmc
auszok@ihmc.us
jbradshaw@ihmc.us



ID	Task Name	Start	Finish	Duration
1	**Ontology Development**	**3/16/2006**	**1/31/2007**	**46w**
2	Design and create a generic COI ontology	3/16/2006	7/4/2006	15.8w
3	Design and create a more specific ontology for COI experimentation	6/7/2006	1/31/2007	34.2w
4	**JBI and KAoS Integration**	**4/10/2006**	**12/25/2006**	**37.2w**
5	Wrap JBI client to enable and enforce semantically-rich client descriptions	4/10/2006	6/30/2006	12w
6	Extend KAoS reasoning mechanisms to exploit the sophistication of the COI ontologies	6/27/2006	12/25/2006	26w
7	Integrate the KAoS Registration Service with JBI	5/18/2006	9/1/2006	15.4w
8	**KAoS and CmapTools Integration**	**4/11/2006**	**12/13/2006**	**35.4w**
9	Develop a new of CmapTools module allowing to observe KAoS registered COI resources, their relations and ontology (in COE)	4/11/2006	10/24/2006	28.2w
10	Extend CmapTool with graphical templates for the generic COI ontology	6/1/2006	12/13/2006	28w
11	**COI policies development**	**9/4/2006**	**1/31/2007**	**21.6w**
12	Develop and demonstrate policies to govern COI management by the IMS personnel	9/4/2006	1/31/2007	21.6w
13	Perform any necessary extensions to the KAoS framework to support policy specification for COI	9/4/2006	11/1/2006	8.6w
14	**COI Demonstration Development**	9/4/2006	2/15/2007	23.8w
15	**Kickoff Meeting**	3/31/2006	3/31/2006	.2w
16	**First Quarterly Review Meeting**	7/6/2006	7/6/2006	.2w
17	**Second Quarterly Review Meeting First Project Demonstration**	10/2/2006	10/2/2006	.2w
18	**Third Quarterly Review meeting**	12/14/2006	12/14/2006	.2w
19	**Final Review Meting - AFRL RRS Third Project Demonstration**	2/28/2007	2/28/2007	.2w

R&D Effort Description

The objective of this effort is to investigate:

(a) methodology and techniques for creating and maintaining COI infospaces;

(b) mechanisms to leverage the KAoS Services framework and COE in order to create a prototype enabling management of semantically-rich COI infospaces;

(c) how these requirements could be supported by demonstration within a JBI testbed

Benefits to the War Fighter

- **Capture and share COI configurations in in two synchronized representations:**
  - **•Graphical concept maps in COE – ease of use**
  - **•Transparent OWL encoding – computer processable**
- **Unique user-friendly COE environment with integrated Web Search, collaboration and version control**
- **Facilitation of the COI implementation through integration with IMS/JBI RI and KAoS**
- **Reuse of COI models**

Technical Challenges

- **Definition of common COI ontology**
- **Enrichment of semantic descriptions of COI infospace**
- **Comprehensive integration of three systems: KAoS, COE, and JBI**
- **Design of the data translation mechanisms**
- **Development of intuitive ontology templates for COI**

Figure 2: Major Phases of the COI Lifecycle Supported by the Prototype

2. Requirement Analysis of Communities of Interest Lifecycle

The initial part of the project was devoted to understanding existing COI research, developing a detailed specification of requirements for each of the COI lifecycle phases, and determining how best to address each of these requirements.

We assembled a sizeable collection of COI resources on the project Web page. From these resources and from discussions with Dr. Larry Warrenfeltz an IHMC expert and former commander at the Naval Oceanographic Office who has extensive experience in the creation of METOC COIs, we derived the requirements below.

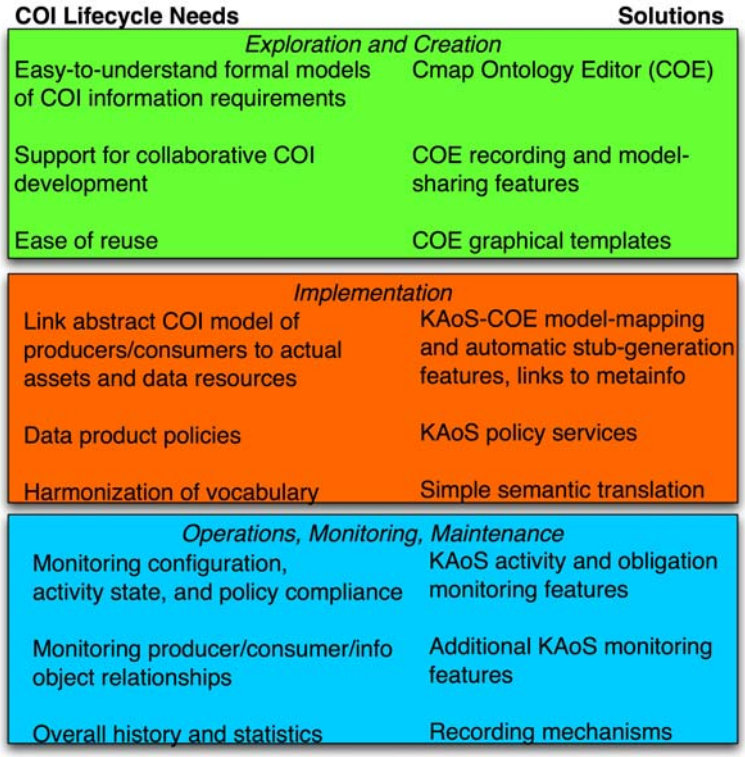


Figure 3: Summary of COI Lifecycle Needs and Solutions

2.1 COI Exploration and Creation Phase Requirements

During the exploration and creation phase, the COI Manager and participants has to define a model for the COI. The important part of this process is agreement on used terms and vocabularies.

Three main requirements were identified:

- *Easy-to-understand formal models of COI information requirements.* The COI-Tool must capture COI requirements and structure in an easy-to-understand format that can be automatically transformed into a formal model of COI information requirements. The Cmap Ontology Editor (COE) allows ontologies to be expressed in graphical Concept Maps rather than in a XML syntax, thus minimizing IMS personnel learning requirements. The tool was adapted to provide contextual filtering of menu items to narrow choices to only these consistent with the current view of the model.

- *Support for collaborative COI development.* To facilitate the process of achieving consensus on COI vocabulary, roles and information to be exchanged, the COI-Tool should provide support for both synchronous and asynchronous distributed collaboration of COI participants. The real-time model-sharing and history recording and playback features of COE address this requirement.
- *Ease of reuse.* COE graphical templates allow individual model elements or larger model structures to be easily extended and reused, thus saving development time in similar COI applications that may later arise. In addition, data from the DoD metadata repository, from Web searches, and WordNet can be easily accessed and reused as starting points in COI development.

2.2 COI Implementation Phase Requirements

This phase of the COI lifecycle involves physical implementation of the COI:

- *Link abstract COI model to implementation model.* Support for this phase must allow mapping and role assignment from the abstract COI model to both physical assets (e.g., military units, individuals) and data resources. Software components must be implemented or adapted to connect the required resources of producers and consumers to the community infospace. Additionally metainformation for community data products has to be easily generated from the model. The COI-Tool prototype provides a graphical interface to support mapping functions. Once the mapping is complete, a COI Manager can use COI-Tool to automatically generate configuration files and software stubs for OIM RI clients that will be used by producers and consumers. Metadata generation is achieved without extra effort in COI-Tool due to the fact that the underlying encoding of COE is already in OWL, permitting the straightforward development of an automated algorithm to map this representation to the XMLSchema needed by OIM RI.
- *Data product policies.* COIs need an expressive and flexible way to define data product access and filtering policies. COI-Tool provides this functionality through its integration with KAoS Policy and Domain Services, which can directly use the OWL ontology of the COI model.
- *Harmonization of vocabulary.* Finally, any realistic COI will need a way to accommodate simple differences in vocabulary among partners within the COI model. COI-Tool provides a prototype of simple semantic translation.

2.3 COI Operation, Monitoring and Maintenance Phase Requirements

During the operational phase the COI Manager must have the capability of:

- *Monitoring configuration, activity state, and policy compliance.* Authorized COI participants must be able to monitor activities inside the community, to discover discrepancies between the model and reality, and to be notified if community policies are violated. COI-Tool addresses these requirements through the integration of OIM RI, existing KAoS Policy Services, and KAoS' new monitoring capabilities. Visualization of monitored data is presented as specialized concept map.
- *Monitoring producer/consumer/info object relationships.* Support for this function is also provided by new KAoS monitoring features.
- *Collecting overall history and statistics.* This requirement is addressed in COI-Tool by adaptation of the COE recording mechanism.

2.4 COI lifecycle information sources and products

The results of our analysis of information flow in the process of creating, implementing and operating the COIs are presented on Figure 4. First, the COI Manager may use different sources of information to bootstrap the process of defining the initial model. Combining specific schemas pulled from the DoD Metadata Repository as well as saved data from previous COIs with the generic COI and KAoS ontologies and results from the integrated Web and WordNet[1] searches, the community model can be rapidly assembled from reusable components and extended to meet new requirements. The model at this stage does not represent any concrete COI but rather a reusable template of the given type of community. For instance, there are many METOC (meteorological) communities in the military that share the same or similar roles and weather data products. So defining a generic template for the METOC community and only refining it for specific case can be a time saver.

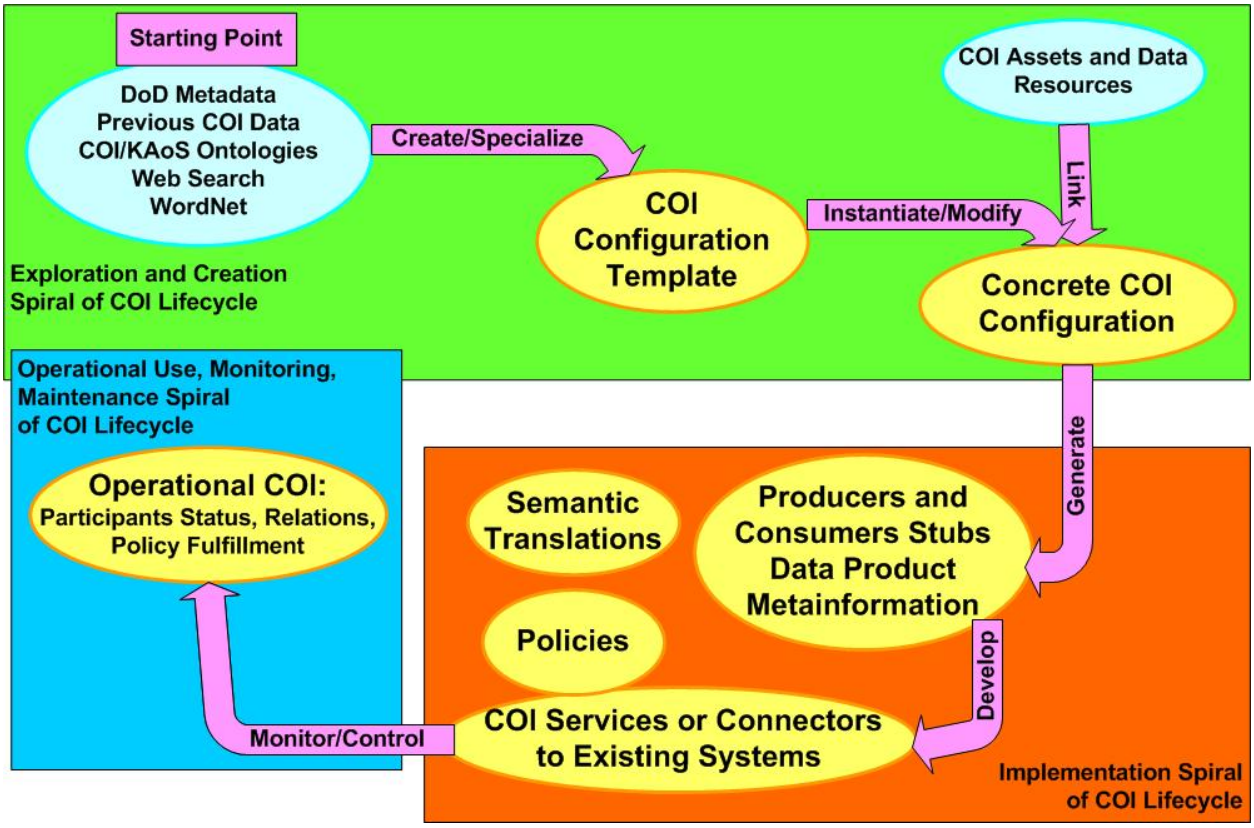


Figure 4: Data products and dataflow of COI lifecycle

The resulting COI Configuration Template contains the following information:

- Links to needed ontologies that specify vocabulary for definition of roles and data product structures,
- Specific producers and consumer types,
- Detailed structure of data products,
- A set of generic policies
- Specific resources such as maps, images, documents, etc.

[1] http://wordnet.princeton.edu/

From the COI Configuration Template, a concrete COI Configuration can be created by mapping producer and consumer roles from the template to resources such as military units, databases, UAVs, and so forth.

Once the available resources have been mapped to assigned roles, software stubs for the implementation phase can be automatically generated. In this phase, additional details must be decided and implemented. For instance, developers responsible for particular consumers or producers will use the stub generated by the COI Manager either to implement new services or connectors to existing ones. The COI Manager can refine (add/change/remove) policies originally defined as part of the generic COI Configuration Template model. As required, simple semantic translations can be defined in order to harmonize data from incompatible participants.

Finally, the clients can now be activated inside infospace. The COI Manager can obtain information about participant status, relations between them and policy fulfillment.

3. Task Objectives and Technical Problems To Be Addressed

During the development phase of the project, our objective was to build and test a prototype implementation of the requirements enumerated during the analysis phase of the project.

In order to address the semantic aspects of COI modeling we based our approach on the most widely-adopted standard today for semantically-rich model representation: the W3C's OWL (Web Ontology Language, http://www.w3.org/TR/owl-features). An exciting part of the results of this project was the first time integration and enhancement of two existing IHMC tools with the AFRL IMS (JBI RI 1.2.6) in order to create a comprehensive environment supporting the COI lifecycle. Here is a brief description of the IHMC tools that we leveraged as part of the project:

- COE (Cmap Ontology Editor, *http://cmap.ihmc.us/coe*) allows OWL ontologies to be conceptualized, developed, and managed though a powerful graphical interface. Unlike similar tools, COE was specifically developed in order to exploit patterns of OWL structure to make it easier for both experts and non-specialists to use (e.g., hiding irrelevant information, use of templates for frequently-used patterns).
- KAoS (*http://ontology.ihmc.us/)* is a robust and mature services framework that relies on OWL in the specification, analysis, and enforcement of policy constraints across a wide variety of distributed computing platforms.

During the project these two tools and AFRL OIM RI had to be integrated for the first time into a coherent system, reusing the same semantics throughout all aspects of its operation. Numerous new mechanisms and capabilities had to been added to these tools to make this possible.

4. Technical Results and COI-Tool Prototype Description

A prototype supporting all three phases of the COI lifecycle was developed. Additional information about this tool and other technical results of the project are available at the project Web site *http://ontology.ihmc.us/COI/*. This Web site contains links to:

- general COI resources,
- METOC COI references,
- COI ontology,

- COI-Tool distribution and installation guide,
- Cmap Server with the demo METOC COI Configuration models and ontologies,
- video tutorials.

Below, we give a detailed description of task performed and functionality developed.

4.1 General Architecture

The tool functionality has been based on ontology. It was assumed that any model of community of interest will start from the generic COI ontology developed during the project, which is based on the KAoS ontology. Through this any community model can be a source of vocabulary for KAoS defined policy. Generic COI ontology captures the basic elements of any COI. Using it as a starting point for domain specific COI ontologies establishes a common ground that will help standardize practice and enable collaboration across COIs. Further to ground the tool to OIM RI mapping between the COI ontology and JBI ontology developed in the J-DASP project (http://jbi.isx.com/jdasp/). COI-Tool produces further layers of specialization to the concrete COI ontologies (Figure 5).

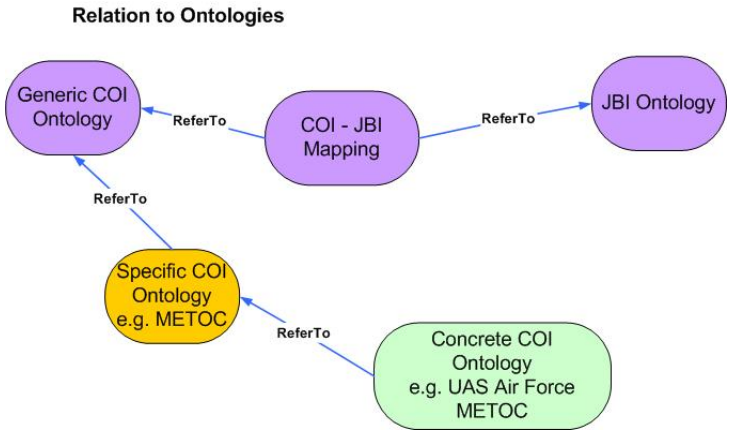


Figure 5: Developed COI ontologies and their relations

Ontology modeling work was a continuous source of requirements for enhancements of the modeling functionality of the COE Ontology Editor.

The COI-Tool is integration of COE, KAoS and OIM RI. COE is used for almost all the graphical interaction with the user with exception of policy definition, which is done by KAoS KPAT interface. We did, however, develop functionality allowing browsing and displaying policies in COE. KAoS provides policy, domain and matching services as well as serving as an integration framework for the COI-Tool, conveying messages between COE and OIM RI clients fulfilling roles in a concrete operating COI.

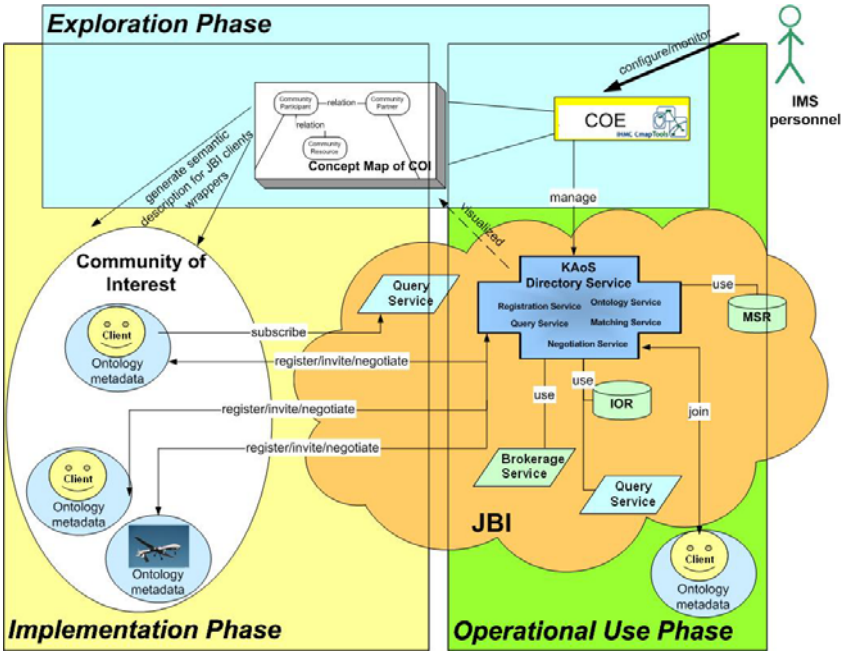


Figure 6: Overview of the COE-KAoS-JBI integration for COI-Tool

4.2 Tasks related to the COI Exploration and Creation Phase of the Lifecycle

This task aimed to provide a rich graphical environment allowing for modeling of the COI Configuration Template and its mapping to the concrete configuration.

Considerable effort was spend on the enhancement of the COE Ontology Editor functionality. When this project started, COE was quite limited in its core ontology modeling functionality and during this project we added the ability for COE to edit and integrate multiple ontologies, an essential requirement for real applications. We also implemented a mechanism to automatically generate OWL encodings of a map and to store these on the CmapServer.

A COI manager is provided with the following functionality when he wants to create a new COI, modify an existing one, or collaborate with other COI managers:

Creation of a new COI template
This generate four COI configuration template placeholders for definitions of community partners (roles/actors), data products, *classes of actions* and COI properties such as information about managers identity, used applications, and so forth. (Fig. 7)

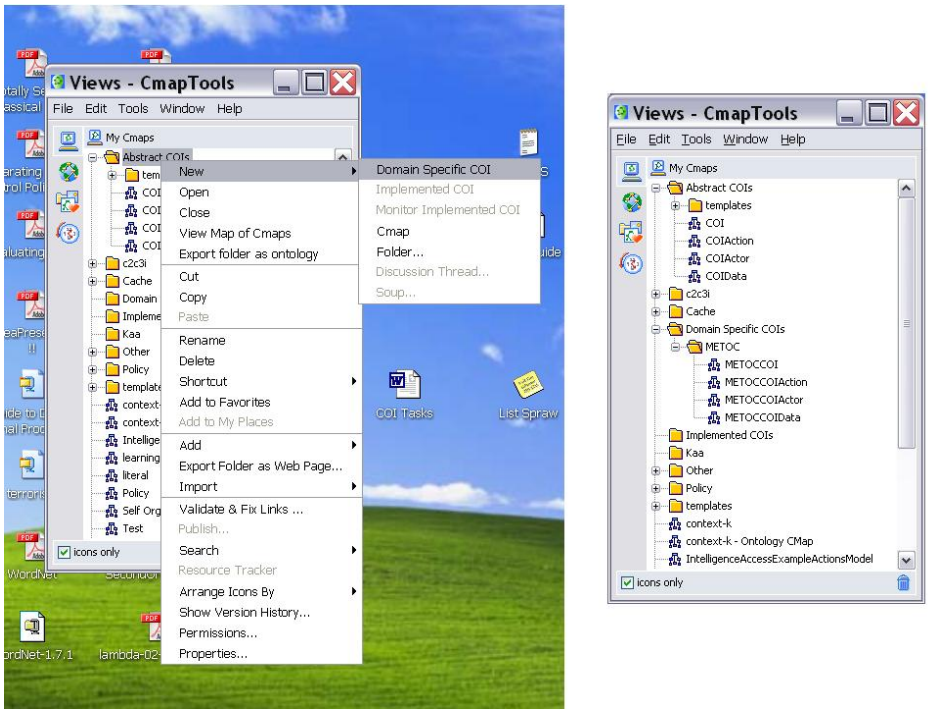


Figure 7: GUI for creation of new domain specific COI and the result

These placeholders are empty concept maps dedicated to particular part of the community model. However they are linked with appropriate part of the COI generic ontology, so the menu of concepts available in the particular maps is narrowed to its context. The new COI template can be started from the generic ontology or from existing COI templates then the new maps are linked to the models of the parent configuration.

Definition of COI properties, Roles and Data Product

A COI manager can open any of these maps in the editor and graphically define concepts and their relations. He can uses the Semantic Space Panel, developed in the scope of this project, to access concepts defined in other concept maps (ontologies) such as for instance the weather ontology when defining weather forecast products (Fig. 8).

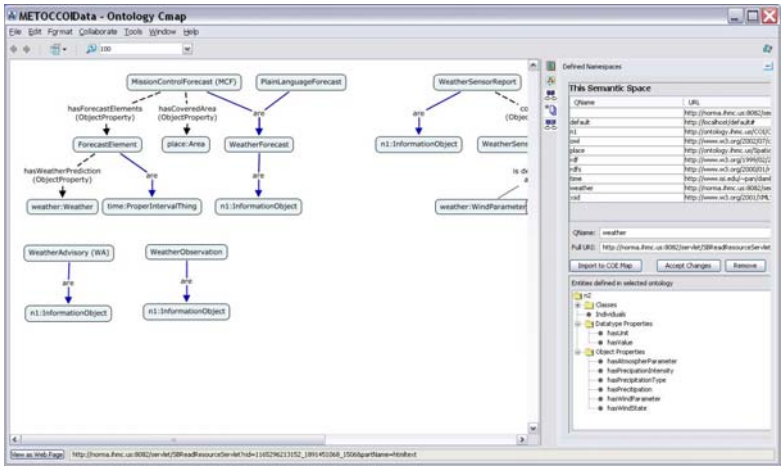


Figure 8: GUI allowing to model COI roles and products

The Semantic Space (Namespace) Panel functionality includes the ability to:

– Show a list of concepts, from the selected namespaces, (as a hierarchy with additional information) applicable to a given selected map node,
– Allow the user to drop a selected concept on the map node and automatically create appropriate map semantic constructs,
– Show current semantic information, from the chosen namespaces, about the selected map node,
– Use the Pellet reasoner through Jena to compute the menu list based on the selected concept node.

Usage of Web Search and WorldNet
COI-Tool provides access to Web resources to get concept definitions and related concepts. The concepts found can be easily added to the map by creation of new map nodes using the button from this search window.

Definition of Relationships between COI Roles and COI Products through the use of COE Templates for map elements
In the scope of the project, we developed automatic generation of concept map templates to be used in child maps. When a given community map is saved, a set of templates is generated from them, making it easy to create subclasses or instances of concept from this map. So, for instance, it is easy to specify that a given community role is a specialization of some other role because of the existence of the template. Annotation of every map with the parent ontology allows the tool to select which template should be shown for a given map. In addition there is a set of static templates making creation of generic ontological relation easy. This set can be easy extended by as COE itself is an editor for templates. COI-Tool has a special panel dedicated to template management (Fig. 9)

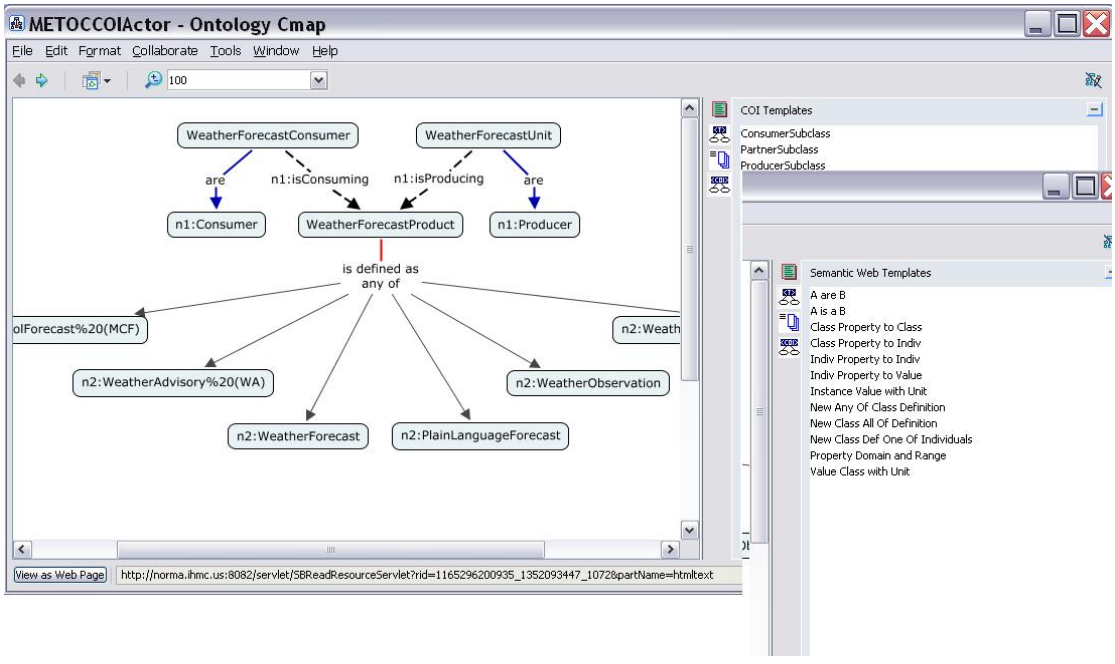


Figure 9: COE template for COI role

Usage of Collaboration Functionality
The COI Manager and his partners are able to simultaneously open and edit a concept map with community vocabulary and models (Fig. 10). They will see other edits and annotations and are

able to send chat messages to themselves. This mechanism allows them to archive **Consensus** on concepts definitions, data product definitions, etc.

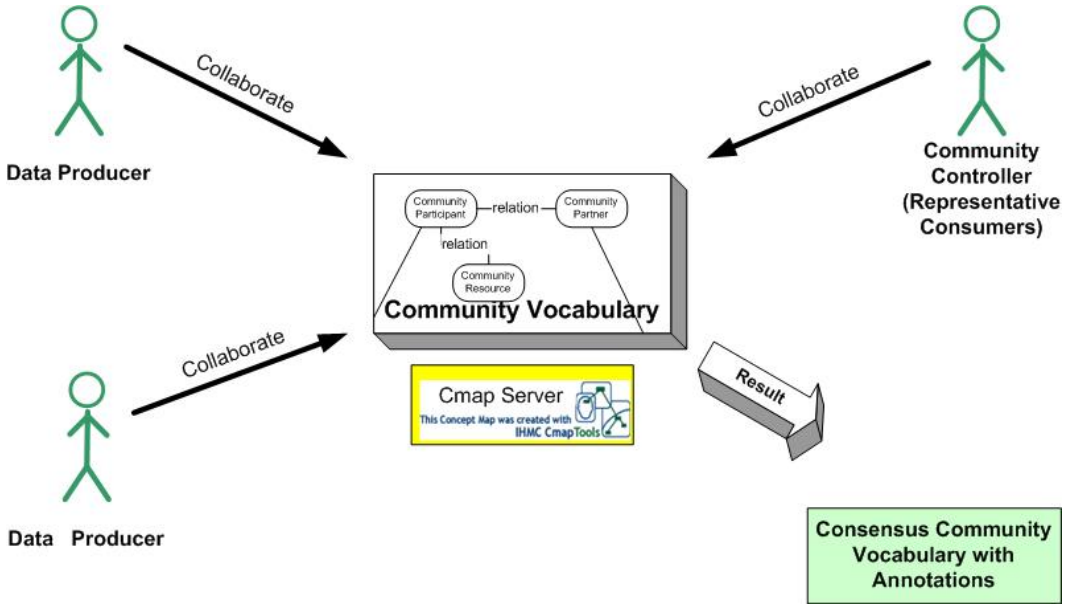


Figure 10: COI-Tool collaboration features

Addition of links and multimedia resources
A COI manager can adds urls to web resource, links to documents and maps to COI concept in order to facilitate their human understanding of the crated community model.

Access to Map historical versions
COE has been integrated with Subversion[2] so it is possible to access previous versions of the developed maps from the integrated version repository.

Creation of a new implemented COI Configuration
A COI manager can creates a implementation of a COI Configuration Template from COI-Tool GUI (Fig 11). The dedicated map for mapping between resources and community roles will have associated templates making this process easier.

[2] http://subversion.tigris.org/

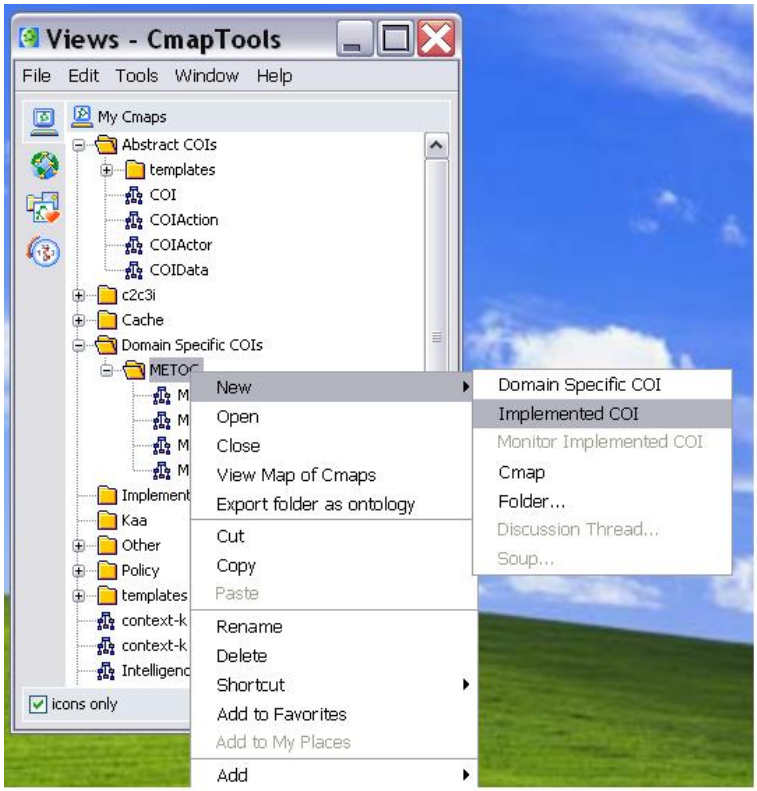


Figure 11: GUI for creation of new implemented COI

4.3 Tasks related to COI Implementation Phase of the Lifecycle

Tasks in this area tried to provide mechanism making software implementation of the defined community easier. The mechanisms developed here are targeted for realization of the COI as cooperating clients in infospheres based on ARFL OIM RI 1.2.6.

Generation of bootstrap files and code skeletons

A COI manger generates a set of files (Fig. 12) for each of the community participant and sends it to the developers in the particular unit. This set contained only the relevant information for the given partner. Based on it the developer can implement OIM RI client providing and consuming given metainformation types and participating in the COI. These files are:

- KAoS agent bootstrap file allowing to the integration of the partner client with the KAoS Policy Service. It containing the following information
  - Actor name,
  - Domain (COI) memberships,
  - Ontology types,
  - List of production channels with the data type of products,
  - List of the subscription channels with the data type of consumed info objects.
- OIM RI client stub file with code opening publish/subscription channels specific to a given partner JBI client, for instance:

  weatherSensorReportSub = initSubscriberChannel(OntologyConcepts.WEATHER_SENSORREPORT_JBI_TYPE, "1.0", ***null****);*

*WeatherWarningChecker callback = **new***
WeatherWarningChecker(weatherSensorReportSub,
OntologyConcepts.WEATHER_SENSORREPORT_CLASS);
weatherSensorReportSub.setSequenceCallback(callback);

- Data type schema files (OWL or mapping to XML Schema),

Figure 12: GUI interface initiating bootstrap file generation

Definition of COI policies

OWL representation of COI configuration defined in COI-Tool can be directly used as ontology vocabulary to define COI policies in KAoS. Policies can be defined using KPAT and during community operation a list of policies applicable to a given partner can be also accessed from COI-Tool. Using KAoS is it not only possible to define authorization policies controlling distribution of information but also obligation policies obliging for instance timely manner of issuing periodic updates.

Definition of semantic translation

In reality, when implementing COI; some partners can use different then agreed in the COI representations of data, for instance when coming from some other community of interest. COI-Tool has been equipped with a prototype graphical interface allowing mapping of structures between two ontological classes defining schemas for data products. The interface generates translation code in SPARQL[3], which then can be used to translate concrete data.

[3] http://www.w3.org/TR/rdf-sparql-query/

4.4 Tasks related to COI Operation, Monitoring and Maintenance Phase of the Lifecycle

In order to provide monitored functionality and other for the community of interest implemented in OIM RI we have developed the interceptor layer for OIM client, which provided the following functionality:

- Partner activation and relation **monitoring**
  - Reports when partner is activated and what data products and from whom it received,
- **Statistical** information about the relation
  - First/last time of data consumption, frequency of consumption, average time between consumption, etc.
- **Authorization** policy checking,
- **Obligation** policy monitoring,
- Semantic **Translation**, if needed.

We renamed OIM RI *client.properties* file to *clientOriginal.properties* and wrote a new client.properties file as below:

```
<capi.implementation>
  <sequences>
  <subscription>coi.jbi.interceptors.KAoSSubscriberSequence</subscription>
  <publication>coi.jbi.interceptors.KAoSPublisherSequence</publication>
  <query>coi.jbi.interceptors.KAoSQuerySequence</query>
  </sequences>
  <connections>
  <connection.manager>
        coi.jbi.interceptors.KAoSConnectionManager</connection.manager>
  <connection>coi.jbi.interceptors.KAoSConnection</connection>
  </connections>
  <repositories>
  <mdr>mil.af.rl.im.capi.client.core.plugin.j2ee.typemgt.MetadataRepository</mdr>
  </repositories>
</capi.implementation>
```

The KAoS interceptor layer for OIM client when initialized reads the *clientOriginal.properties* and forwards the CAPI calls to the original implementation when it performed the functionality listed above. All of it is transparent to the client.

Monitoring of partners activation and relations

The information intercepted in the OIM RI client are forwarded through KAoS to COE if the map with the community model is opened by the manager. The map will show which participants are active and if any unanticipated client exists. It will also show through line any producer–consumer relation and when clicking on the link statistical information such as when the last information was exchange, how many has been exchanged, what is the average time, etc.

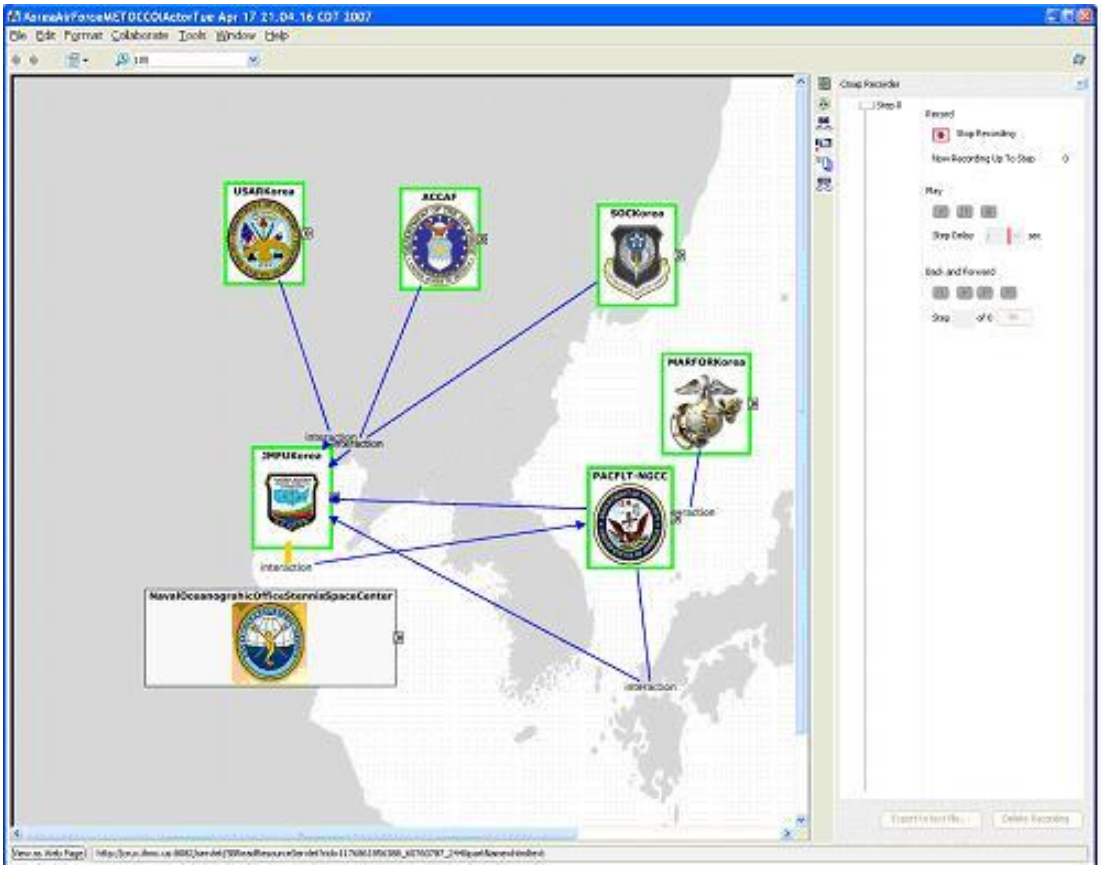


Figure 13: Example of monitored community of interest

Monitoring of obligation policies
In order to monitor producer of information fulfillment of their obligation we have developed a new KAoS mechanism - policy monitor, which allow determining compliance with the obligation policies. If there is violation a feedback loop to the COI Monitor Mode is activated and the notification show up on the map.

Recording of monitoring session and access to recorded session
The recorded history of monitored session can be stored in a special folder associated with the COI configuration. Any recorded session can be later replayed.

4.5 Tasks related to METOC COI Demo

In order to test usability of the developed methodology an example METOC community of interest was selected and developed using the tool. We research this community and collected set of references on http://ontology.ihmc.us/coi/metoc.htm. Using the COI-Tool we developed a weather ontology (Fig 14) and then using it we defined different weather data products as we found in the collected military documents.

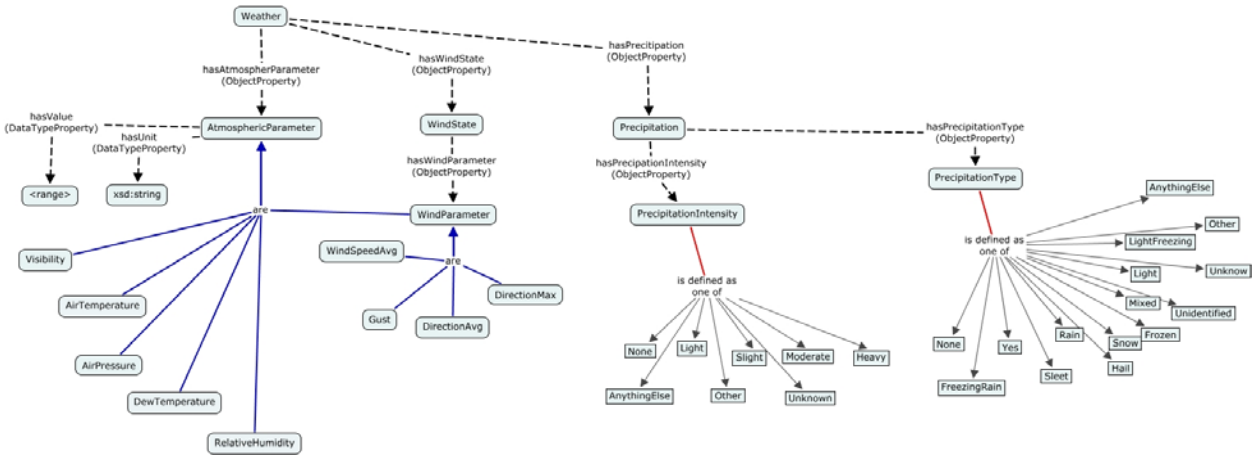


Figure 14: Common METOC vocabulary of weather related concepts

Based on the concrete documents defining the Korea METOC Operation Plan as well as Air and space weather operations, we have defined roles in the community and then mapped them to the actual unities in the Korea military area. Developed models of METOC producers, consumers and their products are available form IHMC CmapServer: http://72.236.182.134:8082/servlet/SBReadResourceServlet?fid=1165296186234_280532411_985. The model was then used to generate bootstrap files and stubs for example clients, which were developed in OIM RI 1.2.6. The clients simulated collection of weather sensor data, distribution of it to the weather forecast centers. The centers were obliged to periodically published weather forecast and if necessary warnings. The example setup of the demo with the components is presented on Figure 15.

This usecase proved that using the tool it is possible to develop the concrete military community of interest and support all the elements of its lifecycle.

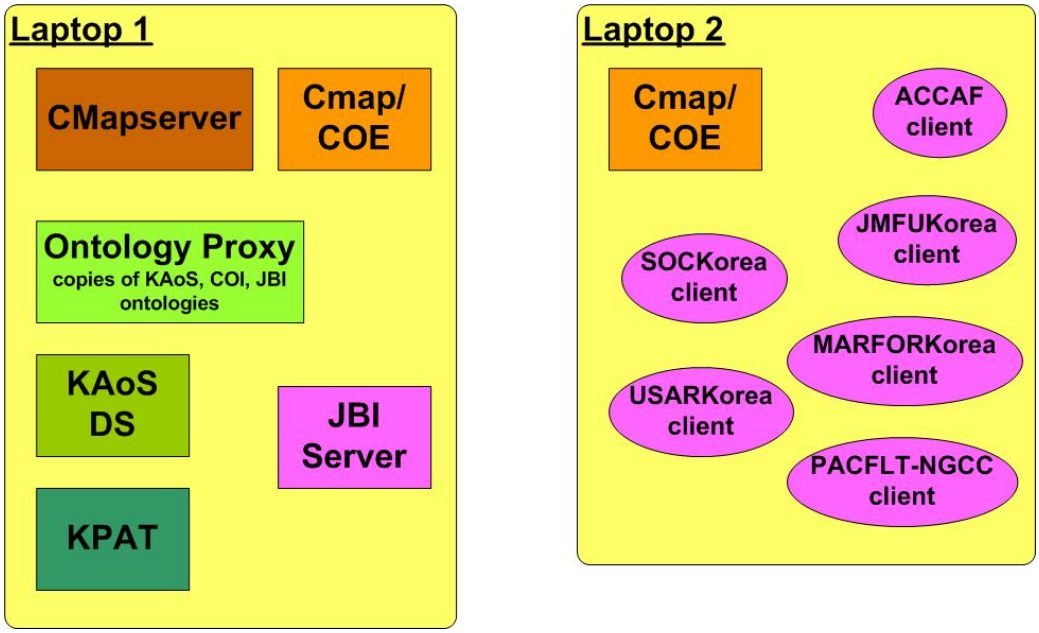


Figure 15: Example Setup and components of the Korea METOC COI demo

5. Conclusion

The important results of the project include a deepened understanding of the community of interest lifecycle, definition of requirements for tools supporting COI lifecycle and description of the COI lifecycle dataflow.

Additionally it was shown how consistent usage of ontology in the COI supporting tool can add significant flexibility and richness to the process.

The developed COI-Tool has these main features

- Capture and share COI configurations in two synchronized representations:
  - Graphical concept maps – easy to use by human,
  - Transparent OWL encoding – computer processable,
- Unique user-friendly Cmap environment with integrated Web Search, simultaneous collaboration and version control,
- Facilitation of the COI implementation through integration with OIM RI and KAoS Policy Service,
- Reuse of COI models.

As any research project also this enumerated many additional features which felled outside the scope of work. Possible and very valuable COI-Tool extensions would be:

- Integration with DoD Metadata Registry through the Web Service interface,
  - to browse and reuse existing Metadata and Taxonomies
  - to contribute developed COI Taxonomies and Schemas (when the DoD Metadata Registry Web Service interface has this functionality)
- Integration with OIM Metainformation catalog to automatically submit COI products metainformation schemas,
- Exploration of support for COI workflow/actions,
- Better support for dynamic communities.

Appendix A: COI-Tool installation guide

The information about the access to the distribution of the COI-Tool is available from the authors of the report.

The COI-Tool consists of few integrated components which require separate installation, preferably in the below sequence:

1. Install in any chosen location the **KAoS Distribution.**

Set up the environment variable *KAOS_HOME* to the directory where the KAoS is installed

2. Install the latest release of the **CmapTools** for Windows (Linux, Mac or Solaris can be made available on request).

When installing chooses option to not install any icons on your desktop or any other place. COI-Tools do not use them and their presents can confuse you in the future.

Set up the environment variable *CMAPTOOLS_HOME* to the directory where the CmapTools is installed

3. Install in any chosen location the **COI-Tools** distribution.

4. Execute the ant script (its default target) from the folder ***scripts/coe-init*** from the COI-Tools installation

5. You can choose from three options to **store models of your COIs**:

- **Local storage** - the same computer on which the tools are installed. If you choose this option you have to execute ant script (its default target) in the folder ***scripts/local-coi-storage-init*** from the COI-Tools installation.

- **Existing COI CmapServer at IHMC**. If you choose this option you have to configure a new Place in CmapTools after you start it. Choose *Places* from the menu on the left and click on the icon in the right corner *Add Place*. In the form choose *Add a place which is not on the list* and use **norma.ihmc.us** (if not reachable please use numeric ip address: **72.236.182.134**) for the *Internet Host Name*, **4447** (default) for the *Port Name* and **8082** for the *Web Server Port Name*. This should add a new place **IHMC COI** to your list of places in CmapTools.

- **Private CmapServer** on any host in your network. Install the CmapServer for Windows (Linux, Mac or Solaris can be made available on request). Set up the environment variable *CMAPSERVER_HOME* to the directory where the CmapServer is installed**.** Also install COI-Tools installation (see point 4) on the same host and execute the ant script (its default target) in the folder ***scripts/private-server-coi-storage-init*** from the COI-Tools installation.

6. Install the **JBI release 1.2.6 client part** on the same machine

Set up the environment variable *JBI_HOME* to the directory where the JBI is installed.

7. To **run the COI-Tools suite** use the following sequence:

- start KAoS executing the ant script in the folder ***scripts/coi*** with target **run-coi-kaos** from the COI-Tools installation,
- start COI-Tools executing the ant script in the folder ***scripts/coi*** with default target from the COI-Tools installation,
- to run demo COI configuration run ant script in the folder ***scripts/jbi-demo*** from the COI-Tools installation,
- you can connect to and monitor the demo COI of METOC community of interest started using ***scripts/jbi-demo*** by selecting the option *Monitor Implemented COI* on the KoreaAirForceMETOCCOIActor from the **Implemented COIs at IHMC** (see point 5) in your running COI-Tools. If you are asked for the KAoS http address please update the ip address in the following url to the one matching the computer on which you running KAoS DS: ***http://127.0.0.1:8081/examples/servlet/TCPServlet*** and provide it to CMapTools.

8. To enable **storing and reviewing of COI configuration history** do the following:

Prerequisites:

Install Subversion version 1.3.0 or later. Note: Ensure your subversion install includes the Java HL client adapter; otherwise, you can download and install the java adapter separately.

Setup:

- Create an empty SVN repository (using the file-based repository type, not BerkleyDB). Sample command: svnadmin *create --fs-type fsfs "c:/path-to-subversion/subversion/svnrepo"*
- Checkout the empty repository to either:
  - 'My Cmaps' for a client install or 'serverRootFolder' for a server install.
  - Server sample command: svn checkout *"file:///c:/path-to-subversion/subversion/svnrepo" "c:/path-to-root/serverRootFolder"*
  - Client sample command: *svn checkout "file:///c:/path-to-subversion/svnrepo" "c:/path-to-mycmaps/My Cmaps"*
  - Note: Do not commit the contents of the serverRootFolder or 'My Cmaps' to Subversion manually. When the server or client starts-up it will automatically update the Subversion repository with the contents of the root folder and all subfolders.
- Edit the configuration settings to enable version control and set the Subversion location information.
  - Server file: serverconfig.txt
  - Client file: cmaptools.cfg
- Enable SVN version control. audit.svn.enabled=true
- Set the URL to the Subversion repository *audit.svn.repository.trunk.url=file\://C\:/path-to-subversion/subversion/svnrepo*
- Add the subversion/bin/ and subversion/javahl/ paths via VM *arg -Djava.library.path* or *LD_LIBRARY_PATH* environment variable